Visvesvaraya Technological University

Belagavi, Karnataka-590 018

A MINI PROJECT REPORT

On

‘Arduino based smart car parking’

Submitted

In partial fulfillment requirements for the award of the Degree

of

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

by

Vaibhav M Karkera 4NM21IS198

Gaurav Shenoy K 4NM21IS216

Devi Kumari 4NM21IS218

Under the Guidance of

Ms. Vanishree B S

Assistant Professor

Department of Information Science and Engineering

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified)

Accredited with ‘A’ Grade by NAAC

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified)
Accredited with 'A' Grade by NAAC

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that Mr.**Vaibhav M Karkera** (4NM21IS198), Mr.**Gaurav Shenoy K** (4NM21IS216) and Ms.**Devi Kumari (**4NM21IS218) has satisfactorily completed the Internet of Things Mini Project work entitled "**Arduino based smart car parking**" of Third Year Bachelor of Engineering in Information Science and Engineering at NMAMIT, Nitte in the academic year 2023 - 24.

Project Guide	**Head, Dept. of ISE**
Ms. Vanishree B S	**Dr. Ashwini B**
Assistant Professor	**Associate Professor**

ABSTRACT

The increasing urbanization and growing population have led to a surge in the number of automobiles, exacerbating parking management challenges in cities worldwide. While traditional parking systems have been studied extensively, smart parking systems have emerged as a promising solution to address modern demands and requirements. This research aims to develop and implement a smart parking system utilizing mobile application technology to effectively manage parking access and enhance security. The proposed system utilizes Arduino UNO, a servo motor, IR sensors, and an LCD display to detect vehicle movements, control entry/exit, and display real-time parking availability information.

Currently, smart parking or parking guidance systems just receive available parking spot information from deployed sensor networks and then simply distribute it to drivers. We'll utilize an Arduino UNO, a servo motor, IR sensors, and an LCD display in this project. IR sensors detect vehicles entering and exiting parking lots and transmit a signal to Arduino; an LCD shows the number of parking spaces available. We can know whether parking spots are available or not even before entering the parking area with the use of such a solution, and we can save gasoline, time, and pollution by using this strategy

TABLE OF CONTENTS **Page no.**

1. Introduction

A smart car parking system is a process through which car parking can be done more efficiently and easily than the manual method. The system will provide the user with better services. The system counts the number of cars in the garage and checks if there's any vacancy. There's an entry and exit path. When a vehicle enters, the display shows the number of cars inside. When any vehicle leaves, the count decreases and is shown on display. If the garage is full. The display will show a message regarding that. This whole process includes the use of Arduino, Display, IR sensor.

2. Components and Modules

In this section, various components, and modules being used for IoT-based Visitor Counter project development are discussed:

2.1 Arduino Uno

Figure 1: Arduino Uno

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced with various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), and 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts.

2.2 LCD display

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly, instead using a backlight or reflector to produce images in color or monochrome. LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images with low information content, which can be displayed.

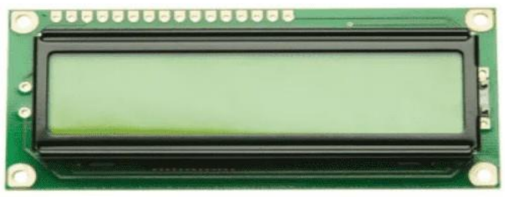


Figure 2: LCD Display

2.3 IR sensor

An infrared sensor (PIR sensor) is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view. They are most often used in PIR-based motion detectors. PIR sensors are commonly used in security alarms and automatic lighting applications. PIR sensors detect general movement but do not give information on who or what moved. For that purpose, an imaging IR sensor is required. PIR sensors are commonly called simply "PIR", or sometimes "PID", for "passive infrared detector".

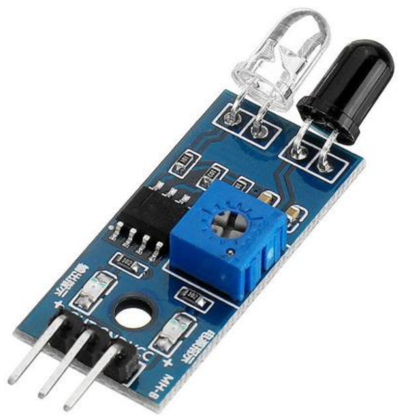


Figure 3: IR sensor

2.4 I2C Module

I 2C (Inter-Integrated Circuit, eye-squared-C), alternatively known as I2C or IIC, is a synchronous, multi-controller/multi-target (controller/target), packet-switched, single-ended, serial communication bus invented in 1982 by Philips Semiconductors. It is widely used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication.

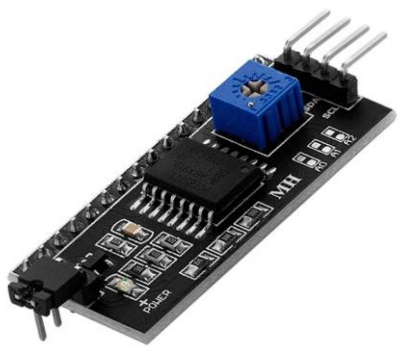


Figure 4: I2c module

3. Circuit description & Working principle

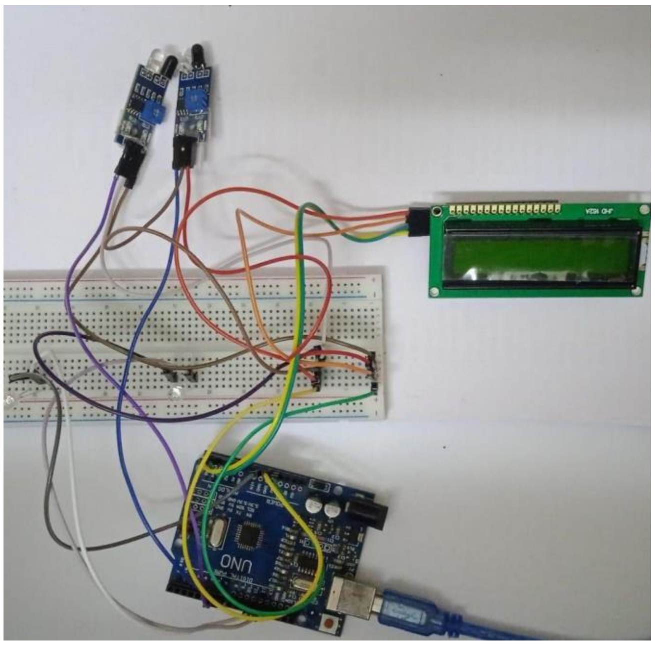


Figure 5: Smart car parking design

In this Arduino-based parking management system, the functionality is divided among several components seamlessly working together. Two infrared (IR) sensors, strategically positioned in distinct parking slots, continuously monitor for the presence of cars. The output from these sensors is processed within the Arduino sketch, and the corresponding parking slot status is visually communicated through dedicated LEDs. When a sensor detects a car, the respective LED illuminates, signifying that the parking spot is occupied. Conversely, a vacant spot is indicated by the LED being turned off.

The system incorporates a LiquidCrystal_I2C display, and a 16x2 LCD screen, to provide a detailed visual representation of the parking status. This display is updated dynamically, reflecting real-time changes in occupancy. Additionally, the I2C protocol streamlines the communication between the Arduino and the LCD screen.

To enhance the system's versatility and accessibility, Blynk integration is employed. Blynk, a platform for building IoT applications, allows users to remotely monitor and control the parking system through a dedicated app. The SoftwareSerial library facilitates communication with Blynk, using virtual pins to receive and respond to commands from the Blynk Terminal widget. Users can input commands such as "parking" to elicit specific responses from the system, enhancing the interactivity of the parking management setup.

In essence, this parking management system combines hardware elements such as IR sensors, LEDs, and an LCD display with the power of online connectivity through Blynk. The result is an efficient, interactive solution that not only monitors parking status but also enables users to interact with the system remotely, providing a seamless and intelligent parking experience.

4. Algorithms & flowchart

4.1 Algorithm

Step 1: Initialize the system

1.1. Initialize the I2C communication bus.

1.2. Initialize the serial communication for debugging and Blynk.

1.3. Initialize the Blynk connection using the auth token.

1.4. Initialize the LCD display and set the backlight on.

1.5. Set the cursor position to the first line of the LCD display.

1.6. Print "Parking Status:" to the first line of the LCD display.

1.7. Set the pin modes for the sensors and LEDs as input and output, respectively.

Step 2: Read sensor inputs

2.1. Read the status of the first IR sensor (car1Status).

2.2. Read the status of the second IR sensor (car2Status).

Step 3: Update parking availability information

3.1. If car1Status is HIGH, print "slot1: vacant" to the LCD display and turn on LED1.

3.2. If car1Status is LOW, print "slot1 occupied" to the LCD display and turn off LED1.

3.3. If car2Status is HIGH, print "Slot 2: vacant" to the LCD display and turn on LED2.

3.4. If car2Status is LOW, print "Slot 2: occupied" to the LCD display and turn off LED2.

Step 4: Send parking status information to Blynk

4.1. Send the parking status information (car1Status and car2Status) to the Blynk IoT platform.

Step 5: Repeat steps 2-4 every second

5.1. Add a delay of 1 second to update the display and send Blynk updates every second.

4.2 Flowchart

The smart parking system utilizes Arduino Uno, IR sensors, LCD display, and the Blynk IoT platform to effectively manage parking availability. Initially, the system establishes communication with the LCD display and sets up serial communication for debugging and sending data to Blynk. It then connects to the Blynk IoT platform and initializes the LCD display, including setting the cursor position and displaying "Parking Status:" on the first line. Finally, it configures the pin modes for the IR sensors as inputs and for the LEDs as outputs.

In a continuous loop, the system reads the status of the IR sensors and stores the values in variables. It then checks the status of each sensor: if a sensor indicates a vacant slot, the system displays the corresponding slot as vacant and turns on the corresponding LED. Otherwise, it displays the slot as occupied and turns off the LED. The system then sends the updated parking status information, including the status of both sensors (car1Status and car2Status), to the Blynk IoT platform. To ensure a controlled update rate, a 1-second delay is inserted before restarting the loop. This continuous monitoring and update process ensures that the system provides real-time parking availability information to drivers.

Flowchart

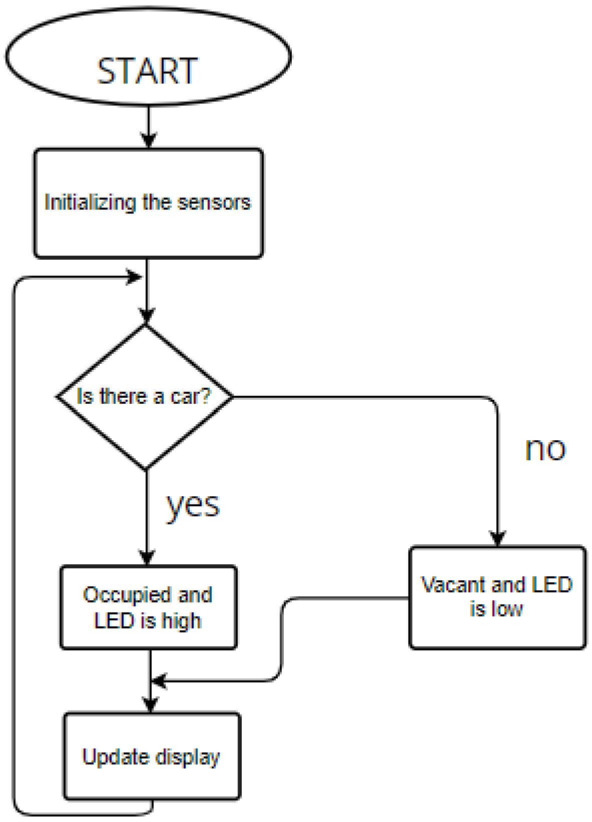


Figure 5: Flowchart of overall process

5. Implementation

During the implementation phase, the theoretical concepts were translated into operational functionalities, ensuring the achievement of all outlined objectives.

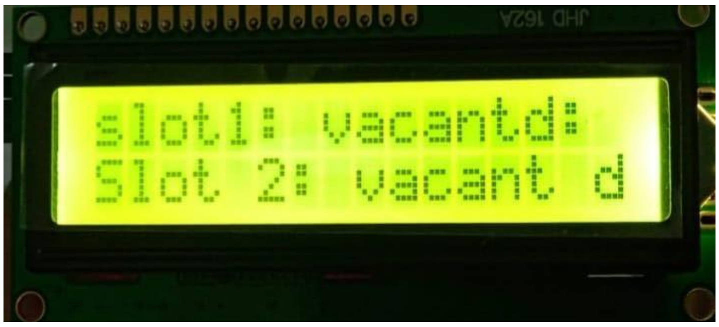
Code:

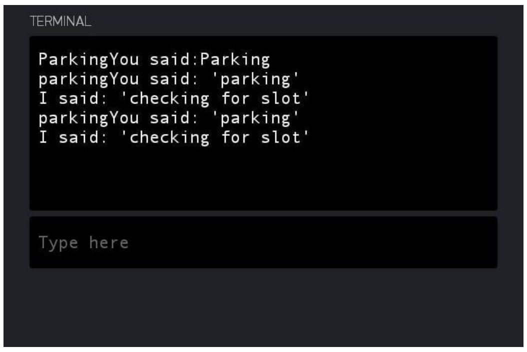
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#define BLYNK_PRINT DebugSerial


// You could use a spare Hardware Serial on boards that have it (like Mega)
#include <SoftwareSerial.h>
```

```
SoftwareSerial DebugSerial(0, 1); // RX, TX

#include <BlynkSimpleStream.h>
WidgetTerminal terminal(V10);
char auth[] = "33GFPAf3c4HAGktB2NGqeUNbdZf0P_7o";

LiquidCrystal_I2C lcd(0x27, 16, 2); // Define the LCD connections
int led1= 11;
int led2= 12;
const int sensor1Pin = 7; // Connect first IR sensor to digital pin 7
const int sensor2Pin = 6; // Connect second IR sensor to digital pin 6

BLYNK_WRITE(V10)
{
  String paramValue = param.asStr(); // Get the value sent from the widget

  // You can use a conditional statement to handle different values if needed
  if (paramValue == "sensor1") {
    int car1Status = digitalRead(sensor1Pin); // Read the status of sensor1

    // Update the Blynk widget with the status of sensor1
    Blynk.virtualWrite(V10, car1Status);

    // Optionally, you can also update the LCD or perform other actions based on the sensor status here.
  }
}
void setup() {
  lcd.init();
   lcd.backlight();  // Initialize the LCD
  lcd.setCursor(0, 0);
  lcd.print("Parking Status:");
  pinMode(sensor1Pin, INPUT);
  pinMode(sensor2Pin, INPUT);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  DebugSerial.begin(9600);

  // Blynk will work through Serial
  // Do not read or write this serial manually in your sketch
  Serial.begin(9600);
  Blynk.begin(Serial, auth);

}

void loop() {

  int car1Status = digitalRead(sensor1Pin); // Read the status of the first sensor
  int car2Status = digitalRead(sensor2Pin); // Read the status of the second sensor

  lcd.setCursor(0, 0); // Set the cursor to the second line

  if (car1Status == HIGH) {
    lcd.print("slot1: vacant");
    digitalWrite(led1, LOW);
```

```
 } else {
  lcd.print("slot1 occupied");
  digitalWrite(led1, HIGH);
 }

 lcd.setCursor(0, 1); // Set the cursor to the third line

 if (car2Status == HIGH) {
  lcd.print("Slot 2: vacant ");
  digitalWrite(led2, LOW);

 } else {
  lcd.print("Slot 2: occupied");
  digitalWrite(led2, HIGH);
 }

  Blynk.run();


}
```

6. Results




TERMINAL

```
ParkingYou said:Parking
parkingYou said: 'parking'
I said: 'checking for slot'
parkingYou said: 'parking'
I said: 'checking for slot'
```

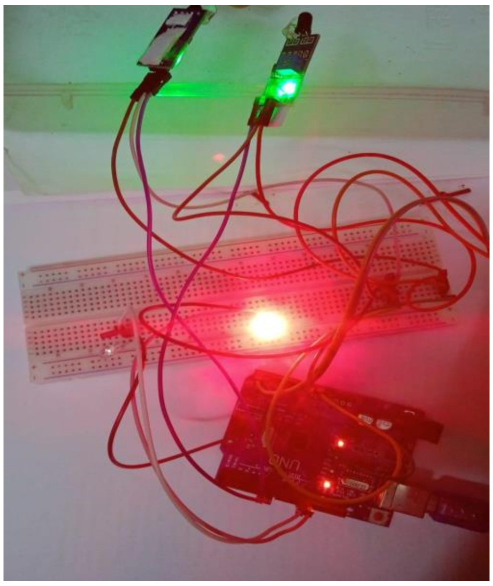
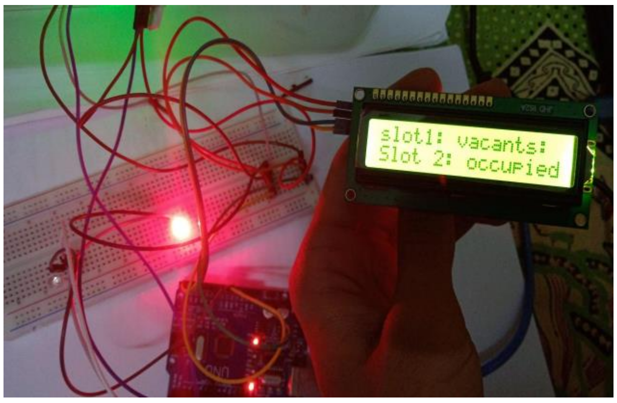



Figure 6: Output

Figure 6 shows the display on the LCD screen if it is vacant or not.

7. Conclusion

The Internet of Things (IoT) was the key concept used to construct the proposed parking system employing an infrared sensor, and this study proposes an effective way to identify a parking space. The IoT-based Car Parking System with an IR Sensor was created as a prototype to help drivers locate a vacant or available parking spot. This parking system employed an infrared sensor to detect the presence and absence of a car to determine the state of a parking slot's availability.

The parking places are continuously monitored, and the data on the LCD screen is updated regularly. The LCD screen shows the exact location of the parking slot availability status. In the meantime, the data from the infrared sensor is also saved in the database. The suggested parking system's prototype was designed for a single storage parking space, but the concept can be expanded to accommodate several storage spaces. In addition, for administrative purposes, a car parking management system interface was created to record the state of a parking slot as well as the precise time a car enters or quits a parking slot. The proposed parking system's conclusion is beneficial for implementation in any parking zone region to assist drivers in finding a vacant parking spot quickly.

8. Future Prospects

1. Integration of smoke sensors for fire detection:

   Enhance safety by integrating smoke sensors into the parking system to detect the presence of smoke and trigger an alarm, alerting authorities and enabling prompt fire suppression efforts.

2. Implementation of ultrasonic sensors or tiny LiDARs for space measurement:

   Improve parking accuracy by incorporating ultrasonic sensors or tiny LiDARs to measure the dimensions of parking spaces, allowing for more precise detection of available slots and enabling the system to accommodate vehicles of different sizes.

References

[1] Google, automatic car parking system project using Arduino by Techatronic
https://techatronic.com/automatic-car-parking-system-project-using-arduino/

[2] Youtube, Arduino Smart Parking System
https://youtu.be/aMRMr4Iu2wY?si=PkhJnCv2Buc_LD3S